Paper Code

AdaProp: Learning Adaptive Propagation for Graph Neural Network based Knowledge Graph Reasoning

*Yongqi Zhang, *Zhanke Zhou, Quanming Yao, Xiaowen Chu, Bo Han

Contact: zhangyongqi@4paradigm.com, cszkzhou@comp.hkbu.edu.hk

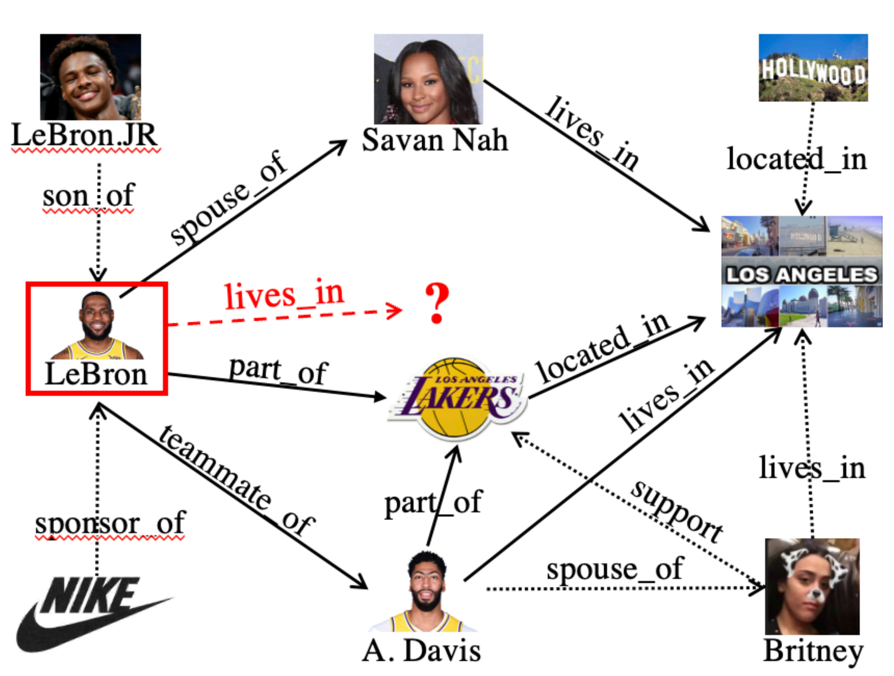
4Paradigm 第四范式 · 香港浸會大學 HONG KONG BAPTIST UNIVERSITY · 清華大學 Tsinghua University · 香港科技大學(廣州) THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

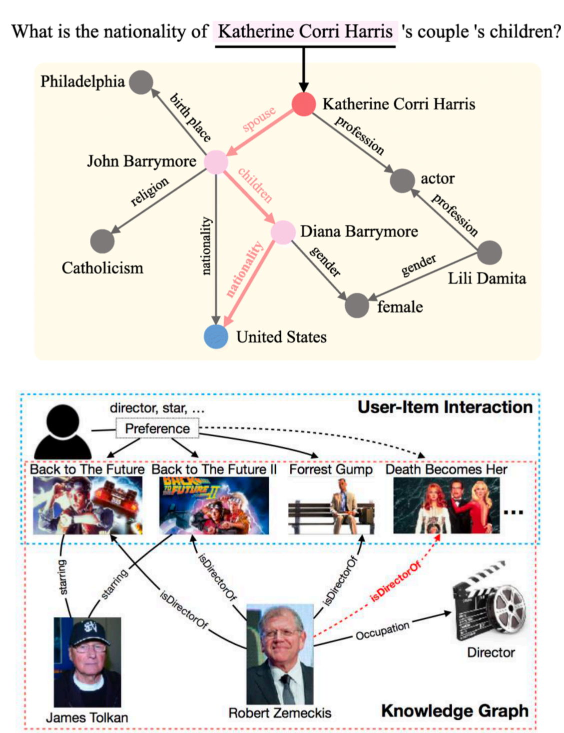
TL;DR: An important design component of GNN-based KG reasoning methods is called the propagation path, which contains a set of involved entities in each propagation step. Existing methods use hand-designed propagation paths, ignoring the correlation between the entities and the query relation. In addition, the number of involved entities will explosively grow at larger propagation steps. In this work, we are motivated to learn an adaptive propagation path in order to filter out irrelevant entities while preserving promising targets.

Background: KG Reasoning

KG reasoning with query: (LeBron, lives_in, ?)

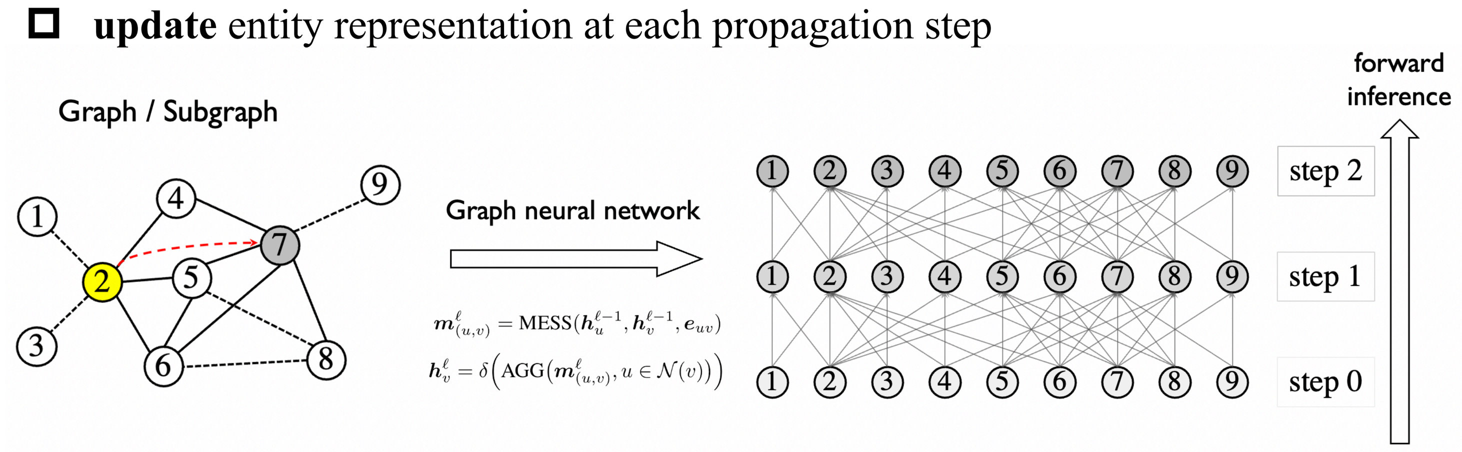
Applications: QA / Recommendation





Graph Neural Network-based methods for KG reasoning

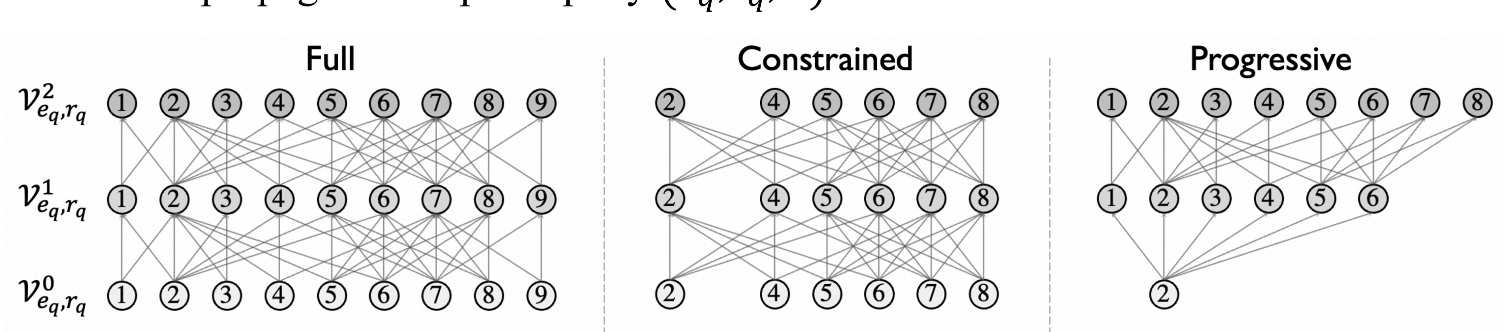
- ☐ **propagate** the message with the graph structure
- ☐ **update** entity representation at each propagation step



The Propagation Path

Query-dependent propagation path $\widehat{\mathcal{G}}^L_{e_q,r_q}$

- ☐ $\widehat{\mathcal{G}}^L_{e_q,r_q} = \{\mathcal{V}^0_{e_q,r_q}, \mathcal{V}^1_{e_q,r_q}, \dots, \mathcal{V}^L_{e_q,r_q}\}$ as the sets of involved entities
- ☐ in each propagation step for query $(e_q, r_q, ?)$



Problems when L is large

- ☐ *Full* propagation: large memory cost & over-smoothing
- ☐ *Constrained* propagation: extremely high inference cost
- ☐ *Progressive* propagation: exponentially increased nodes

Problem & Challenges

Problem formulation: Reduce the size of propagation path through **sampling**

$$\widehat{\mathcal{G}}^L_{e_q,r_q} = \{\mathcal{V}^0_{e_q,r_q}, \mathcal{V}^1_{e_q,r_q}, \dots, \mathcal{V}^L_{e_q,r_q}\},$$

$$\text{s.t. } \mathcal{V}^\ell_{e_q,r_q} = \begin{cases} \{e_q\} & \ell = 0 \\ S(\mathcal{V}^{\ell-1}_{e_q,r_q}) & \ell = 1 \dots L \end{cases}.$$

Two challenges of the sampling strategy $S(\cdot)$

- ☐ the target answer e_a is unknown given $(e_q, r_q, ?)$
- ☐ semantic dependency is complex

Existing sampling approaches are not applicable

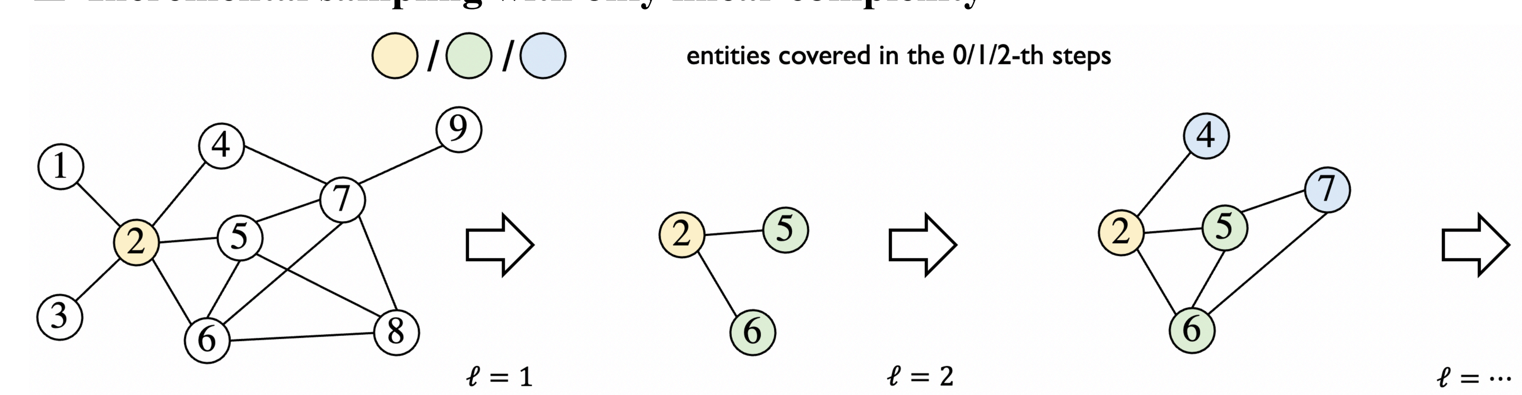
- ☐ no target preserving
- ☐ no relation consideration
- ☐ no direct supervision

Method: adaptively sample semantically relevant entities during propagation

Design1: Connection-preserving Incremental Sampling

☐ **Key idea:** Preserve the previous entities & sample from the newly visited ones $\quad \mathcal{V}^0_{e_q,r_q} \subseteq \mathcal{V}^1_{e_q,r_q} \cdots \subseteq \mathcal{V}^L_{e_q,r_q}$

☐ **Incremental sampling with only linear complexity**



☐ **Details in each step: Candidate generation and sampling**

Candidate generation: the newly-visit neighboring entities of last step

$$\overline{\mathcal{V}}^\ell_{e_q,r_q} := \text{CAND}(\mathcal{V}^{\ell-1}_{e_q,r_q}) = \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q}) \setminus \mathcal{V}^{\ell-1}_{e_q,r_q}.$$

e.g. ① ③ ④ ⑤ ⑥ when $l = 1$
① ③ ④ ⑦ ⑧ when $l = 2$

Candidate sampling: sample K entities without replacement from candidates

$$\mathcal{V}^\ell_{e_q,r_q} := \mathcal{V}^{\ell-1}_{e_q,r_q} \cup \text{SAMP}(\overline{\mathcal{V}}^\ell_{e_q,r_q}).$$

e.g. ⑤ ⑥ when $l = 1$
④ ⑦ when $l = 2$

Design2: Learning-based and Sematic-aware Distribution

☐ **Key idea:** Introduce a parameterized distribution & borrow knowledge from the GNN $\quad \mathcal{V}^\ell_{e_q,r_q} = S(\mathcal{V}^{\ell-1}_{e_q,r_q}; \boldsymbol{\theta}^\ell)$

Parameterized sampling distribution:

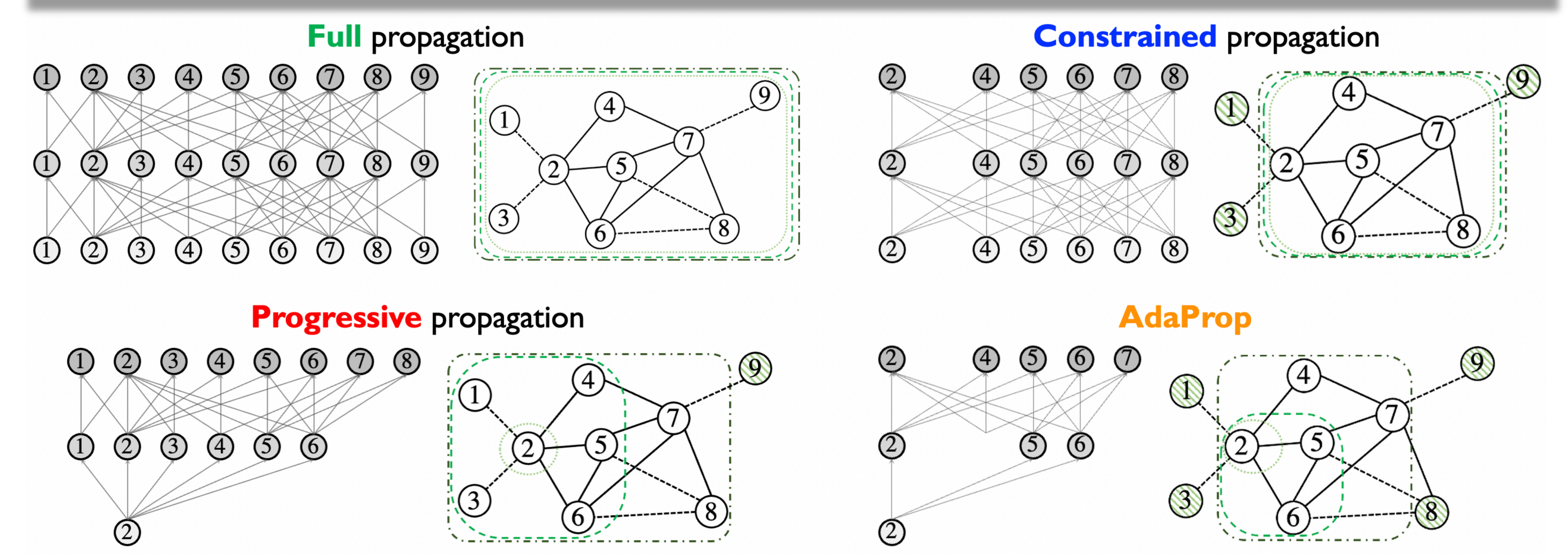
- ☐ Sharing the knowledge in GNN representations $\boldsymbol{h}^\ell_e$
- ☐ Adaptive based on the learnable parameters $\boldsymbol{\theta}^\ell$

$$p^\ell(e) := \exp\left(g(\boldsymbol{h}^\ell_e; \boldsymbol{\theta}^\ell)/\tau\right) \Big/ \sum\nolimits_{e' \in \overline{\mathcal{V}}^\ell_{e_q,r_q}} \exp\left(g(\boldsymbol{h}^\ell_{e'}; \boldsymbol{\theta}^\ell)/\tau\right)$$

Learning strategy:

- ☐ Gumbel-trick to enable backward propagation on hard samples.
- ☐ Sampling: get top-K based on gumbel-logits
  $G_e := g(\boldsymbol{h}^\ell_e; \boldsymbol{\theta}^\ell) - \log(-\log U_e)$ with $U_e \sim \text{Uniform}(0,1)$ for the candidate entities
- ☐ Enable backpropagation: straight-through estimation
  $\boldsymbol{h}^\ell_e = \left(1 - \text{no\_grad}(p^\ell(e)) + p^\ell(e)\right) \cdot \boldsymbol{h}^\ell_e$ for the selected entities

An overall comparison with existing propagation schemes



Adaprop achieves smaller complexity, deeper steps and query-dependent.

Comprehensive Experiments

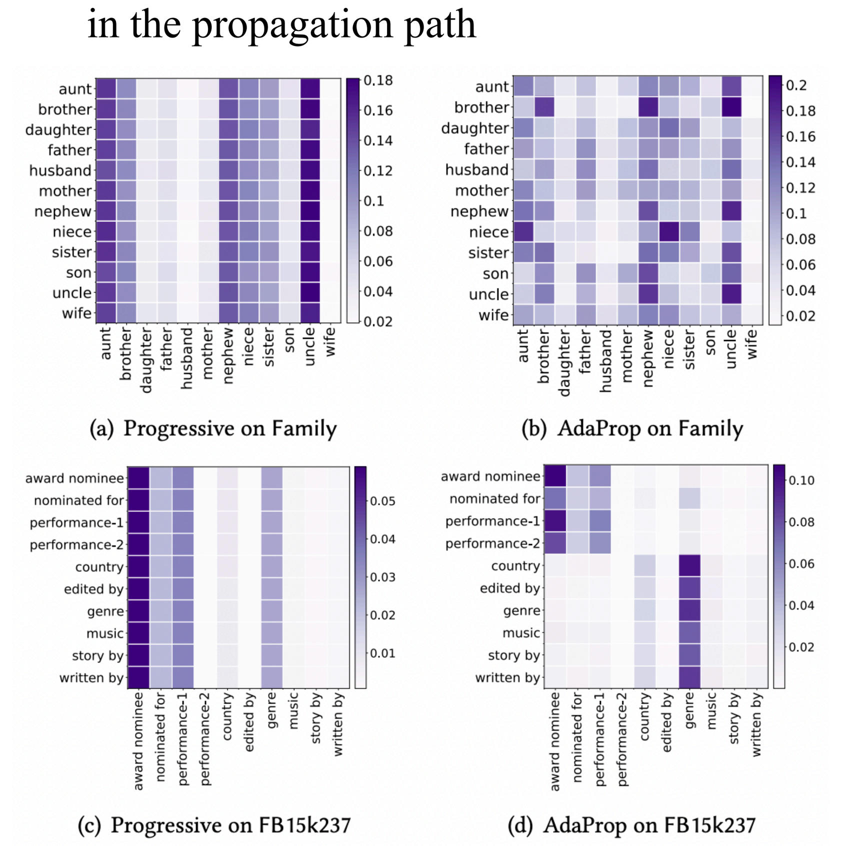
☐ Evaluation with transductive settings

type	models	Family MRR	Family H@1	Family H@10	UMLS MRR	UMLS H@1	UMLS H@10	WN18RR MRR	WN18RR H@1	WN18RR H@10	FB15k237 MRR	FB15k237 H@1	FB15k237 H@10	NELL-995 MRR	NELL-995 H@1	NELL-995 H@10	YAGO3-10 MRR	YAGO3-10 H@1	YAGO3-10 H@10
non-GNN	ConvE	0.912	83.7	98.2	0.937	92.2	96.7	0.427	39.2	49.8	0.325	23.7	50.1	0.511	44.6	61.9	0.520	45.0	66.0
	QuatE	0.941	89.6	99.1	0.944	90.5	99.3	0.480	44.0	55.1	0.350	25.6	53.8	0.533	46.6	64.3	0.379	30.1	53.4
	RotatE	0.921	86.6	98.8	0.925	86.3	99.3	0.477	42.8	57.1	0.337	24.1	53.3	0.508	44.8	60.8	0.495	40.2	67.0
	MINERVA	0.885	82.5	96.1	0.825	72.8	96.8	0.448	41.3	51.3	0.293	21.7	45.6	0.513	41.3	63.7	–	–	–
	DRUM	0.934	88.1	<u>99.6</u>	0.813	67.4	97.6	0.486	42.5	58.6	0.343	25.5	51.6	0.532	46.0	66.2	0.531	45.3	67.6
	RNNLogic	0.881	85.7	90.7	0.842	77.2	96.5	0.483	44.6	55.8	0.344	25.2	53.0	0.416	36.3	47.8	0.554	50.9	62.2
	RLogic	–	–	–	–	–	–	0.47	44.3	53.7	0.31	20.3	50.1	–	–	–	0.36	25.2	50.4
GNNs	CompGCN	0.933	88.3	99.1	0.927	86.7	<u>99.4</u>	0.479	44.3	54.6	0.355	26.4	53.5	0.463	38.3	59.6	0.421	39.2	57.7
	NBFNet	0.989	**98.8**	98.9	0.948	92.0	**99.5**	<u>0.551</u>	<u>49.7</u>	<u>66.6</u>	<u>0.415</u>	<u>32.1</u>	**59.9**	0.525	45.1	63.9	0.550	47.9	68.6
	RED-GNN	**0.992**	**98.8**	**99.7**	<u>0.964</u>	<u>94.6</u>	99.0	0.533	48.5	62.4	0.374	28.3	55.8	<u>0.543</u>	<u>47.6</u>	<u>65.1</u>	0.559	48.3	68.9
	AdaProp	<u>0.988</u>	98.6	99.0	**0.969**	**95.6**	**99.5**	**0.562**	**49.9**	**67.1**	**0.417**	**33.1**	<u>58.5</u>	**0.554**	**49.3**	**65.5**	**0.573**	**51.0**	**68.5**

☐ Evaluation with inductive settings

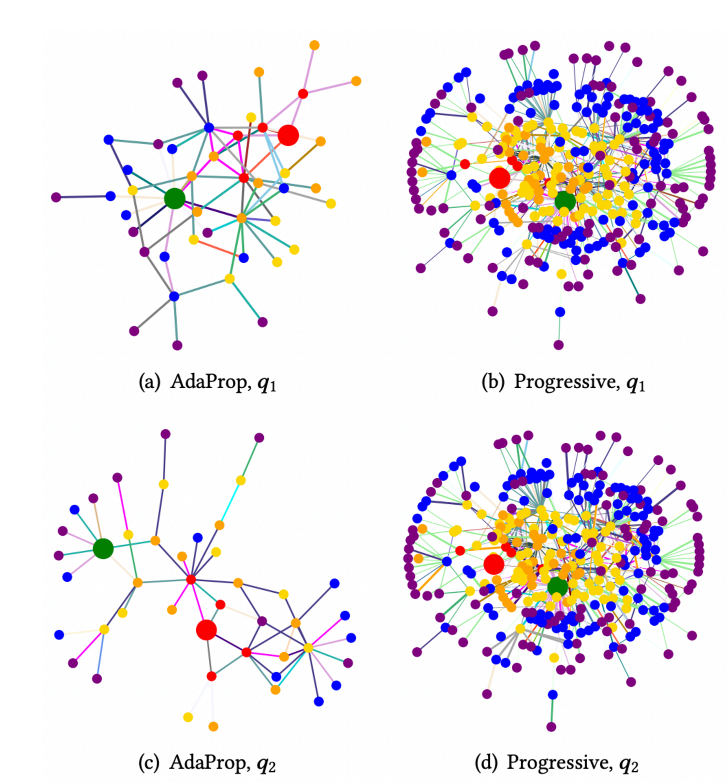
metric	methods	WN18RR V1	WN18RR V2	WN18RR V3	WN18RR V4	FB15k237 V1	FB15k237 V2	FB15k237 V3	FB15k237 V4	NELL-995 V1	NELL-995 V2	NELL-995 V3	NELL-995 V4
Hit@10 (%)	RuleN	73.0	69.4	40.7	68.1	44.6	59.9	60.0	60.5	76.0	51.4	53.1	48.4
	Neural LP	77.2	74.9	47.6	70.6	46.8	58.6	57.1	59.3	87.1	56.4	57.6	53.9
	DRUM	77.7	74.7	47.7	70.2	47.4	59.5	57.1	59.3	<u>87.3</u>	54.0	57.7	53.1
	GraIL	76.0	77.6	40.9	68.7	42.9	42.4	42.4	38.9	56.5	49.6	51.8	50.6
	CoMPILE	74.7	74.3	40.6	67.0	43.9	45.7	44.9	35.8	57.5	44.6	51.5	42.1
	NBFNet	<u>82.7</u>	<u>79.9</u>	<u>56.3</u>	70.2	<u>51.7</u>	<u>63.9</u>	58.8	55.9	79.5	<u>63.5</u>	<u>60.6</u>	<u>59.1</u>
	RED-GNN	79.9	78.0	52.4	<u>72.1</u>	48.3	62.9	60.3	<u>62.1</u>	86.6	60.1	59.4	55.6
	AdaProp	**86.6**	**83.6**	**62.6**	**75.5**	**55.1**	**65.9**	**63.7**	**63.8**	**88.6**	**65.2**	**61.8**	**60.7**

☐ Heatmaps of relation type ratios in the propagation path



semantic-aware

☐ Exemplar propagation paths on FB15k237-v1 dataset



connection-preserving